第3章

FPGA開発ツールで シミュレーションを体験する <<Xilinx 編>>

― ISE Simulator活用チュートリアル

ここでは，米国Xilinx社のFPGA開発ツール「ISE」が標準で搭載しているシミュレーション機能「ISE Simulator」を活用する方法を解説する．CPLDや機能モジュールのように小規模な検証を手軽に行うことができる．本誌付属DVD-ROMには，無償で利用できるISE Simulator Liteの機能を持つISE WebPACKを収録している[編集部注]．　（編集部）

米国Xilinx社のFPGA開発ツール「ISE」に，バージョン7.1iから論理シミュレータが搭載されました．これは「ISE Simulator」と呼ばれています．無償版の「ISE WebPACK」には，15,000行までのHDLソース・コードに対応する機能限定版の「ISE Simulator Lite」が付属します．フル機能のISE Simulatorは，フル機能の製品版ツール「ISE Foundation」のオプションとして提供されています．

1．ISE Simulatorの特徴

FPGA開発の際に使用する論理シミュレータとしては，米国Mentor Graphics社の「ModelSim」が有名です．もし，すでにModelSimを使いこなしているのであれば，ISE Simulatorを使う理由はないと思います．ISE SimulatorはModelSimほどの機能はありませんし，大規模なシミュレーションもできません．いたってシンプルな論理シミュレータなのです．

編集部注：本稿はISE WebPACK 8.2iをベースに解説しており，付属DVD-ROMにも同ツールを収録している．最新版の9.1iはXilinx社のWebサイトからダウンロードできる．

● 操作が簡単で小規模の検証を手軽にできる

シンプルなシミュレータなので，操作はいたって簡単です．操作が簡単なので，ISE 7.1i以上がインストールされていて，VHDLまたはVerilog HDLの基本を習得された方なら，すぐに使えます．

ISE Simulatorは，大規模シミュレーションには向きませんから，

- CPLDを開発するとき
- モジュール単位のシミュレーションをするとき

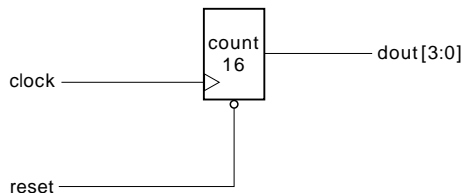
に適すると言えます．特に，モジュール単位のシミュレーションをサクッと行いたい場合には，操作が簡単な分だけ便利なシミュレータと言えます．

● ISE Simulatorで機能を，ISE本体でタイミングを検証

FPGAを開発する際の検証項目としては，論理的な不具合を確認するためのシミュレーションと物理的にFPGA内部の遅延を確認するための検証（STA：static timing analyze）があります．

STAについては，FPGA内部のタイミング・エラーに関してISEが出力するタイミング・レポートで，どこが悪いのか，何が原因なのかについて確認できます．このタイミング・エラーは，どんなに大規模な回路でも遅延時間の大きい順に表示されます．見慣れれば，どこでなにが起きているのかは，レポートを確認することで状況を知ることができます．

KeyWord　FPGA，ISE Simulator Lite，STA，テストベンチ，4ビット・バイナリ・カウンタ，プロジェクト

図1
4ビット・カウンタの入出力仕様
クロックの立ち上がりごとにカウントアップする4ビット・カウンタである．カウント値が4ビットで出力される．reset入力により，カウント値がクリアされる．

(**a**) ブロック図

方向	信号名	説　明
入力	clock	クロック入力
	reset	リセット信号入力
出力	dout[3:0]	カウンタの出力

(**b**) 入出力の概要

リスト1
4ビット・カウンタのソース・コード(count16.v)

```
module count16(clock,reset,dout);
  input clock,reset;
  output [3:0] dout;
  reg [3:0] dout;

always @ (posedge clock or negedge reset)
  begin
    if(~reset)
        dout <= 4'b0000;
    else
        dout <= dout + 1;
    end
  endmodule
```

リスト2
4ビット・カウンタのテストベンチ(tb_count16.v)

```
`timescale 1ns / 1ps

module tb_count16_v;
  reg clock,reset;
  wire [3:0] dout;
  parameter STEP = 100;

  count16 uut (
          .clock(clock),
          .reset(reset),
          .dout(dout));

  initial begin
          clock = 0;
          reset = 0;
  end

  initial #100 reset = 1;

  always #(STEP/2)
       clock = ~clock;

  endmodule
```

● 大規模な設計におけるモジュール単位の検証

検証にはいろいろ方法があると思います．シミュレーションは大規模になればなるほど波形を観測しながら，不具合個所を捜し当てるのに時間がかかるようになります．シミュレーションの実行に時間がかかってしまい，何よりテストベンチの作成が大変になります．

FPGAはプログラマブル・デバイスなので，何か間違いがあったら何度でも書き換え可能です．そこで，実機で動作させながら，不具合個所を探し当てて修正していく方法も考えられます．しかし，この方法では，論理的にVHDL/Verilog HDL記述が悪いのか，物理的にFPGA内部のタイミングが悪いのかといった切り分けができません．

とはいえ，大規模すぎるとシミュレーションによる検証が現実的ではなくなります．このようなとき，モジュール単位でサクサクと検証を進めていくとよいでしょう．テストベンチも単純で済みますし，シミュレーション時間もかかりません．回路規模が小さければ，不具合個所の特定もしやすいはずです．このような場面で，ISE Simulatorが役に立つことでしょう．

不具合の個所は，できるだけ開発作業の初期につぶしておくのがよいと思います．

2．ISE Simulatorによる検証を体験する

ISE Simulatorは，いたってシンプルなシミュレータです．ここでは，小規模な回路をサンプルとして利用して，使い方を解説します．

サンプル回路は，4ビット・バイナリ・カウンタです(**図1**)．ソース・コードを**リスト1**に示します．また，テストベンチを**リスト2**に示します．

解説を進めるにあたって，回路は設計済み(ソース・コードとテストベンチは作業フォルダにすでに用意されている)とします(付属DVD-ROMにも収録されている)．

● プロジェクトの作成

シミュレーションを実行するためには，論理合成や配置配線のときと同様にプロジェクトを作成します(**図2**)．

まず，新規にプロジェクトを作成します．ISEを起動した状態のProject Navigator画面から「File」→「New Project...」を選択すると，New Projectウィザードが始まります．

1) プロジェクトの設定

ウィザードの1ページ目(Create New Project)では，プロジェクトの設定を行います．最初にProject Location欄にプロジェクトを作成するフォルダ名を入力します．ここ

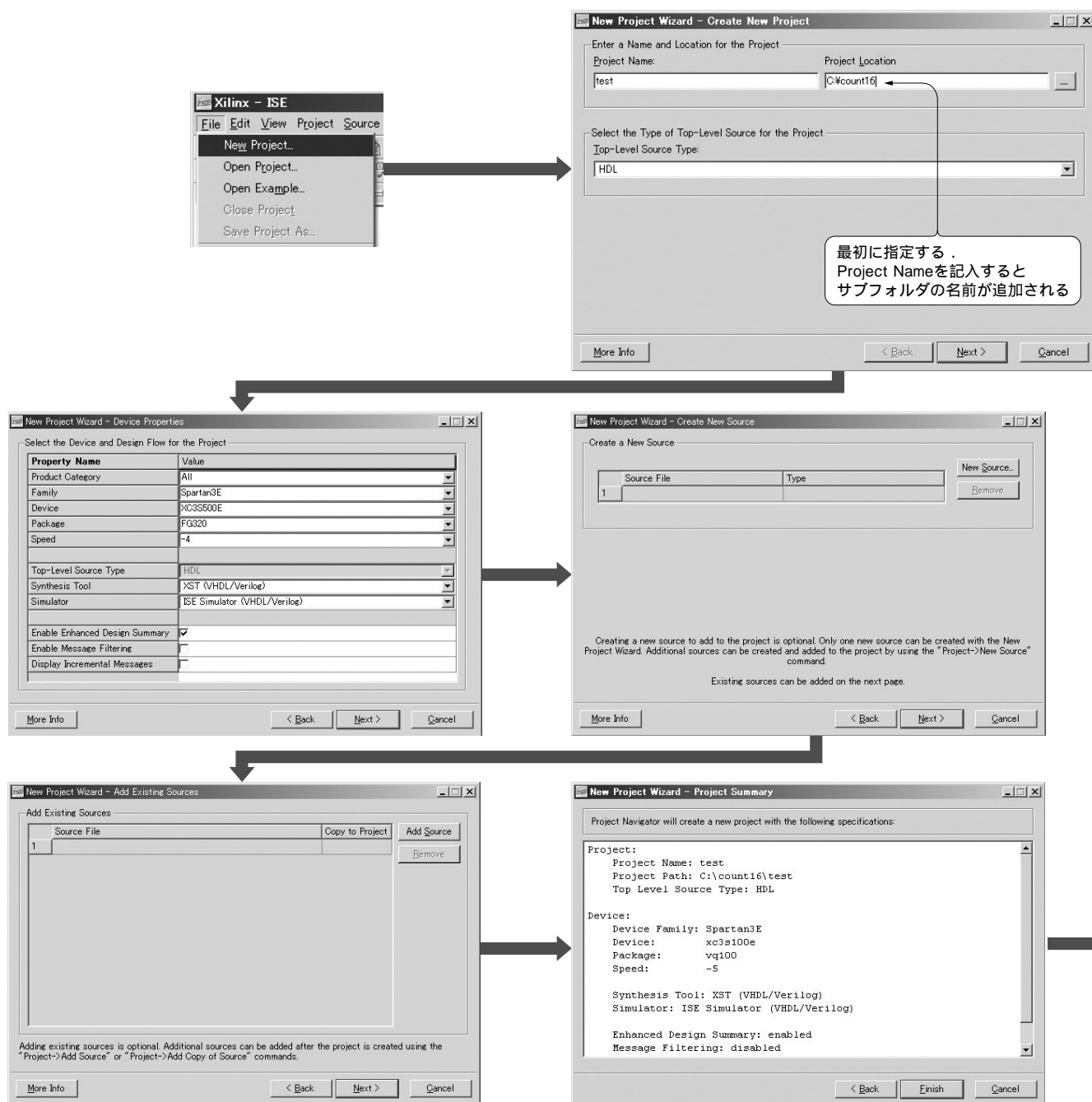



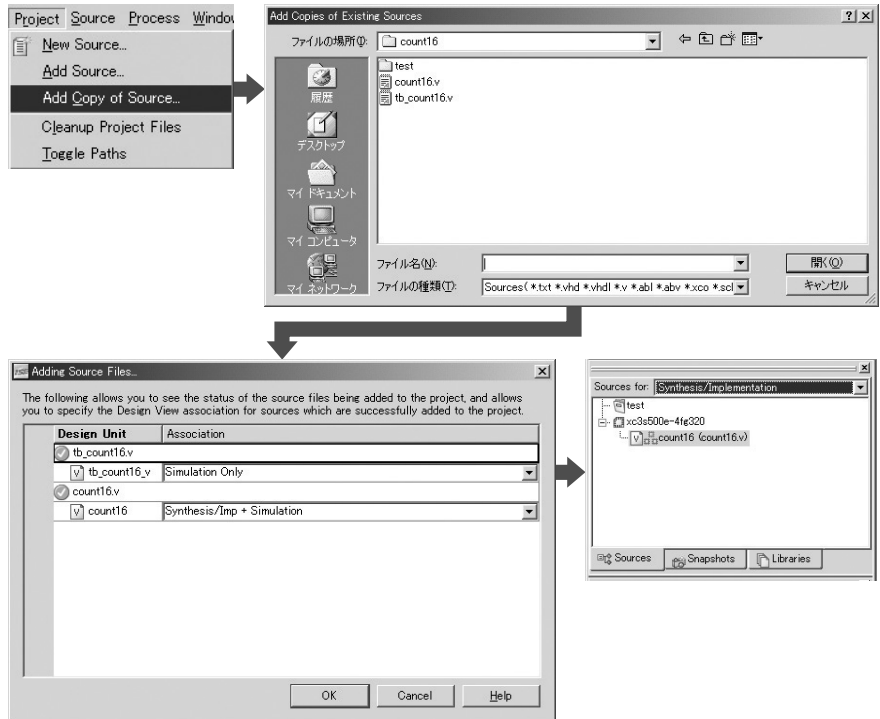
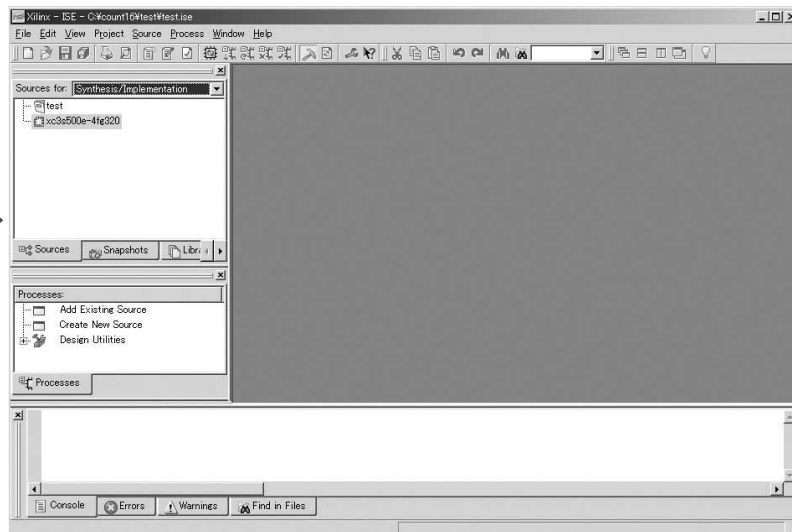
図2　プロジェクトの作成

では「C:\count16」とします．既存のフォルダを指定する場合は，［...］ボタンをクリックして，リストから選択してもかまいません．次に，Project Name欄にはプロジェクト名を入力します．ここでは「test」とします．プロジェクトを作成するフォルダにプロジェクトのサブフォルダが生成されるので，Project Location欄の表示がC:\count16\testに変わっていることを確認します．最後にTop-Level Source Type欄で，プルダウン・メニューの中から「HDL」を選択します．

2）ターゲット・デバイスの指定

2ページ目（Device Properties）では，実際に設計するデバイスを選択します．ここではSpartan-3EファミリのうちXC3S500E-4FG320をターゲットとして，Device Family欄は「Spartan3E」，Device欄は「XC3S500E」，Package欄は「FG320」，Speed Grade欄は「-4」に設定します．使用するシミュレータの指定はここで行います．Simulator欄

図3
ソース・ファイルの追加

3

が「ISE Simulator(VHDL/Verilog)」に設定されていることを確認します．

3) ソース・コードの作成と追加

3ページ目(Create New Source)と4ページ目(Add Existing Sources)は何も設定せずに[次へ]をクリックします．

4) サマリの確認

5ページ目(Project Summary)で設定した内容を確認します．最後に[Finish]をクリックすれば，プロジェクトが作成されます．

● ソース・ファイルの設定

次にソース・ファイルとテストベンチをプロジェクトに追加します(**図3**)．

Project Navigatorのメニューから「Project」→「Add Copy of Sources」を選択します．あらかじめ用意してある

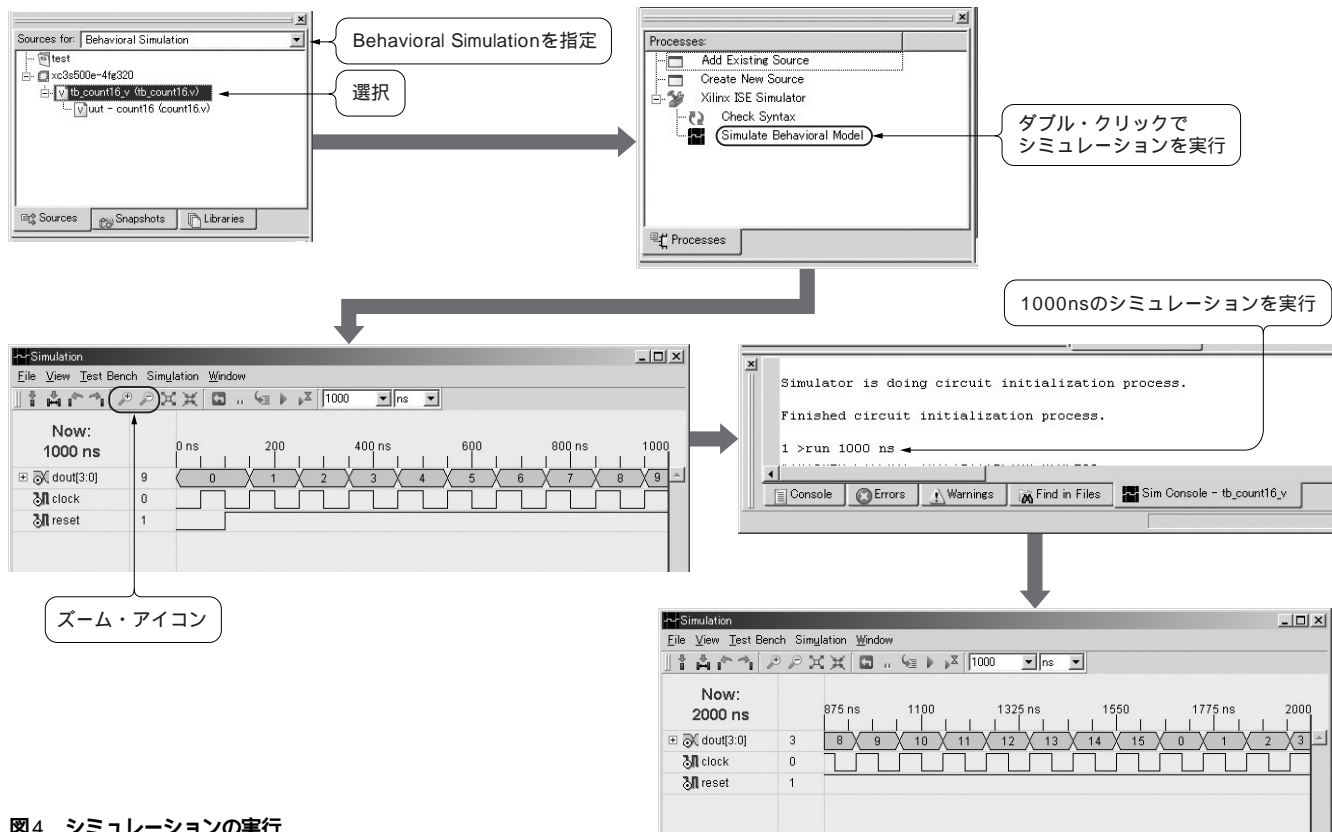



図4　シミュレーションの実行

ソース・ファイルcount16.vとテストベンチtb_count16.v（ここでは作業フォルダのC:\count16に記録されているものとする）を指定します．

Adding Source Fileウィンドウが開きます． count16.vは「Synthesis/Imp+Simulation」，tb_count16.vは「Simulation Only」になっていることを確認します．

これでProject Navigatorの画面で，ファイルが登録されました．

● シミュレーションの実行

いよいよシミュレーションの実行です（**図4**）．

Project NavigatorのSources for:ウィンドウのプルダウンから，「Behavioral Simulation」を選択します．そして，Sources for:ウィンドウでテストベンチ「tb_count16.v」を選択します．

Processesウィンドウで「Xilinx ISE Simulator」の下にある「Simulate Behavioral Model」をダブル・クリックすると，ISE Simulatorがシミュレーションを開始します．

シミュレーションが終わると波形ウィンドウが表示されます．シミュレーション結果を見ながら，動作を確認します．ズームのアイコンを使えば，波形を拡大/縮小できます．

シミュレーションは1000nsで終了しています．さらにシミュレーションを実行するには，メッセージ・ウィンドウで，

```
run（実行すべき時間）
```

を実行します．例えば，さらに1000nsのシミュレーションを実行するには，

```
run 1000 ns
```

とします．これで2000nsまでシミュレーションが実行されます．

いけざわ・よういち
アヴネットジャパン（株）